

FIGURE 1
(Prior Art)

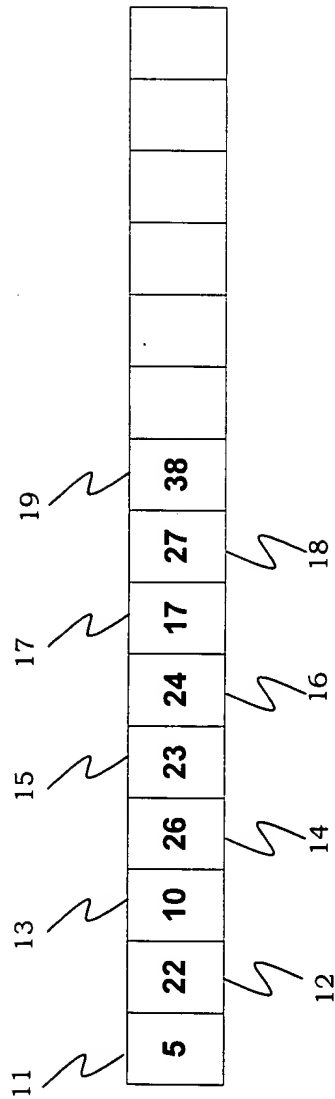
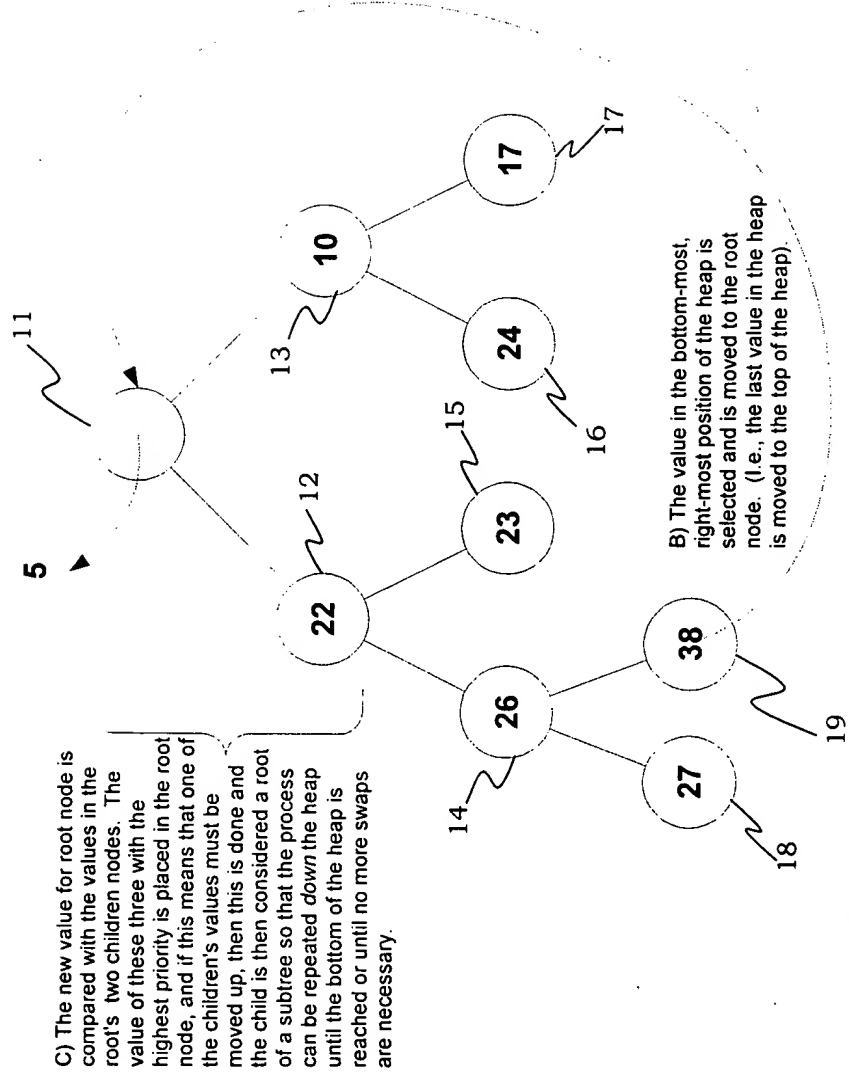


FIGURE 2
(Prior Art)

FIG. 3

A) The value in the root node is removed, leaving a "hole".



C) The new value for root node is compared with the values in the root's two children nodes. The value of these three with the highest priority is placed in the root node, and if this means that one of the children's values must be moved up, then this is done and the child is then considered a root of a subtree so that the process can be repeated down the heap until the bottom of the heap is reached or until no more swaps are necessary.

B) The value in the bottom-most, right-most position of the heap is selected and is moved to the root node. (I.e., the last value in the heap is moved to the top of the heap).

FIGURE 3
(Prior Art)

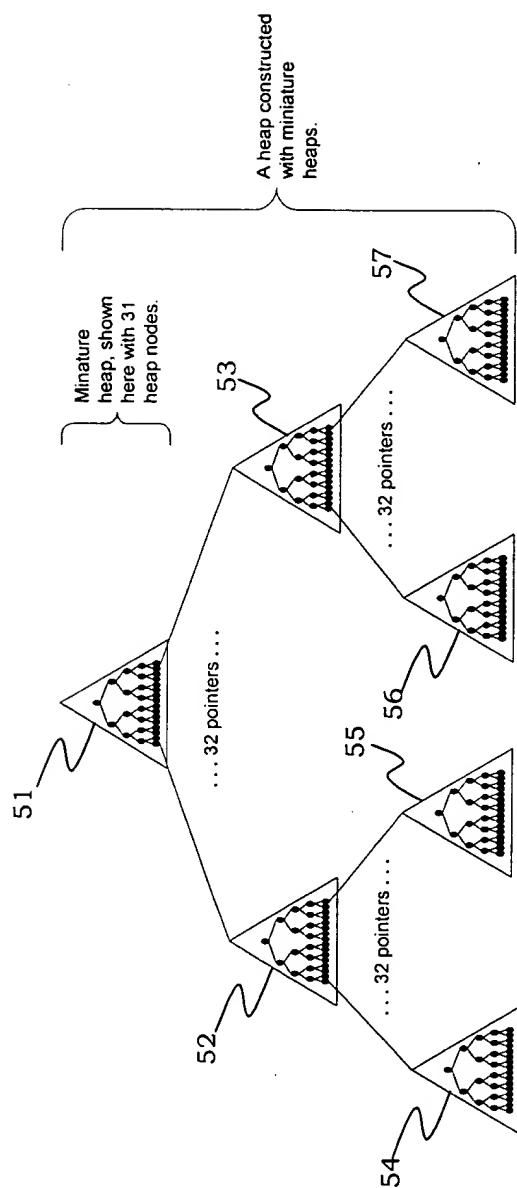


FIGURE 5

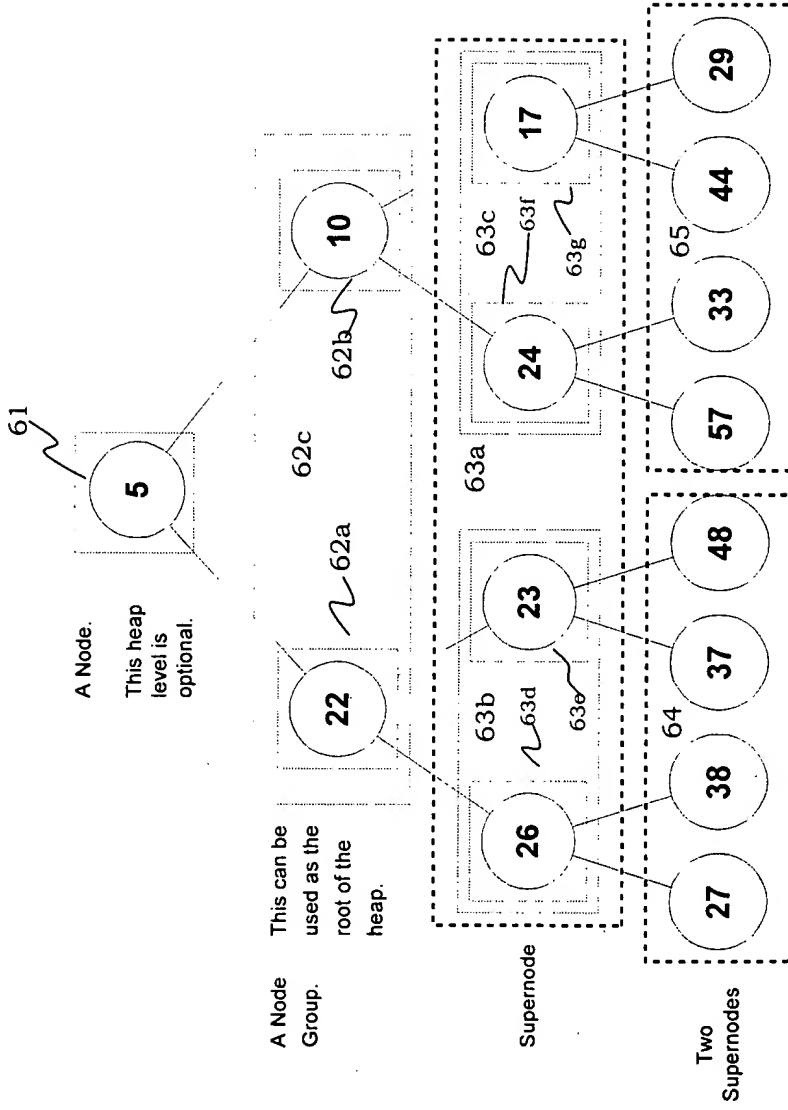


FIGURE 6

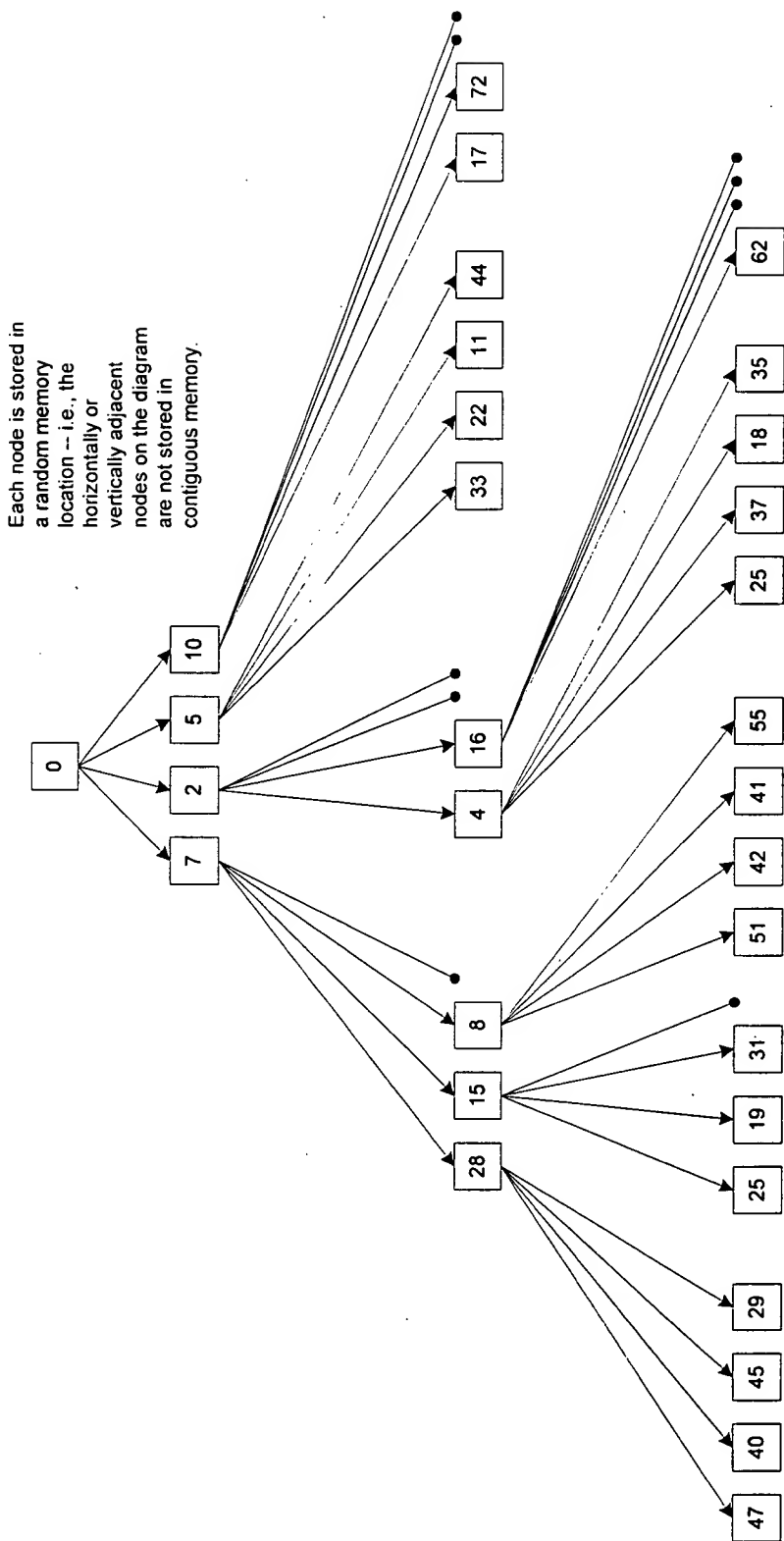


FIGURE 7

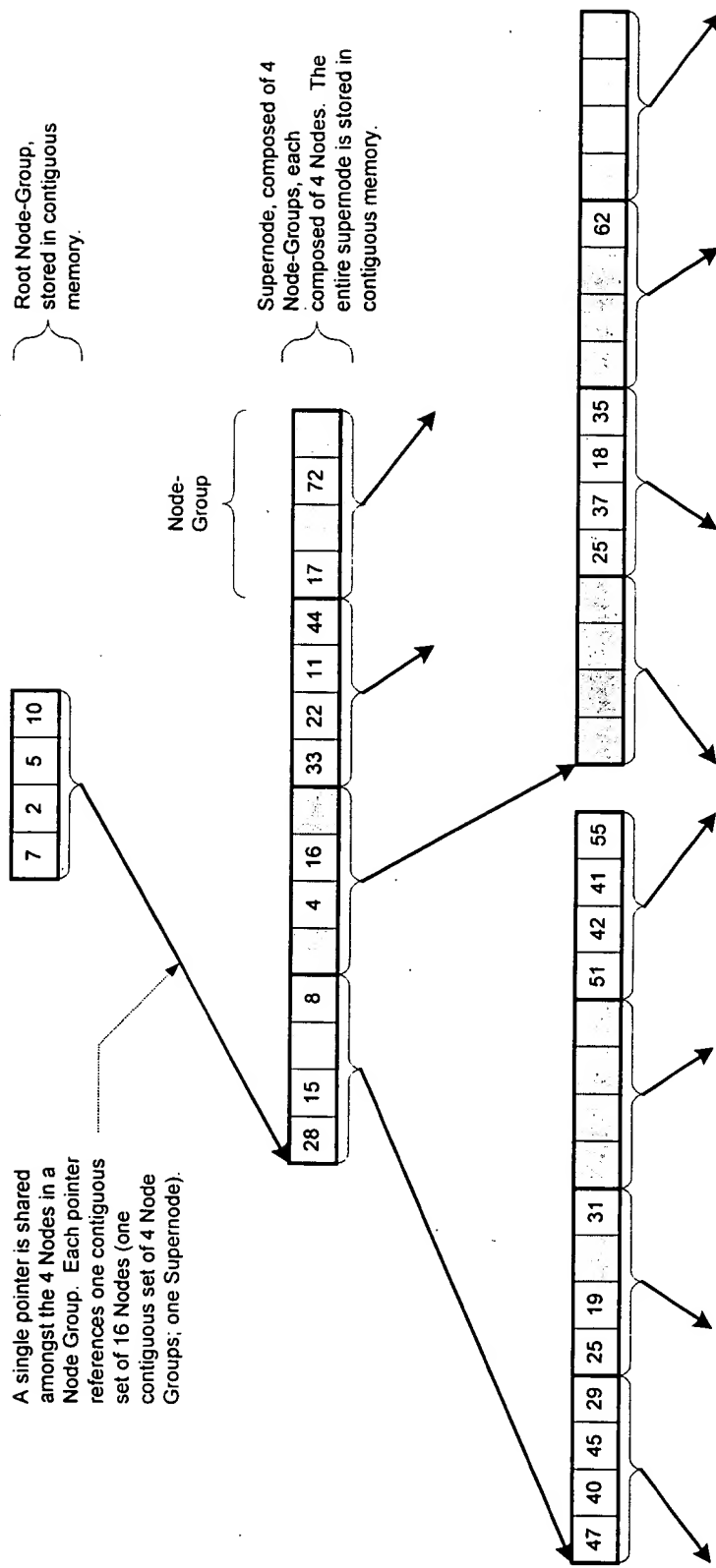


FIGURE 8

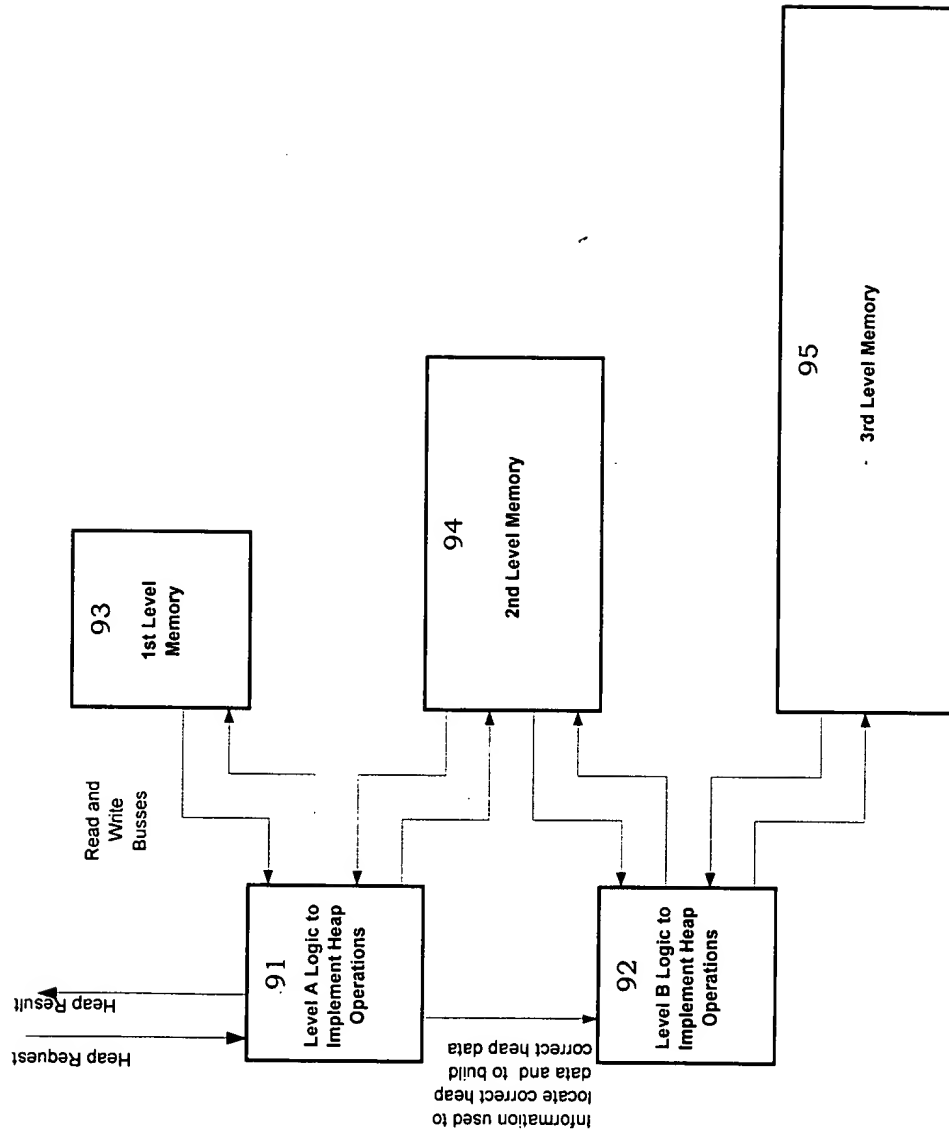


FIGURE 9

[illegible][illegible]

FIGURE 10

FIG. 11 is a block diagram of a multi-level heap management system. The system includes a first level memory (114) and a second level memory (115). A first level insert logic and comparators (111) are connected to the first level memory (114) via read and write busses. A first level remove logic and comparators (117) are connected to the first level memory (114) via read and write busses. A second level insert logic and comparators (112) are connected to the second level memory (115) via read and write busses. A second level remove logic and comparators (118) are connected to the second level memory (115) via read and write busses. A third level memory (116) is connected to the second level memory (115) via read and write busses. A third level insert logic and comparators (113) are connected to the third level memory (116) via read and write busses. A third level remove logic and comparators (119) are connected to the third level memory (116) via read and write busses. The system also includes a heap insert request input and a heap remove request input. The first level insert logic and comparators (111) receive the heap insert request and output information used to locate correct heap data and to build correct heap data to the second level insert logic and comparators (112). The second level insert logic and comparators (112) output information used to locate correct heap data and to build correct heap data to the third level insert logic and comparators (113). The first level remove logic and comparators (117) receive the heap remove request and output information used to locate correct heap data and to build correct heap data to the second level remove logic and comparators (118). The second level remove logic and comparators (118) output information used to locate correct heap data and to build correct heap data to the third level remove logic and comparators (119). The third level insert logic and comparators (113) output information used to locate correct heap data and to build correct heap data to the third level memory (116). The third level remove logic and comparators (119) output information used to locate correct heap data and to build correct heap data to the third level memory (116).

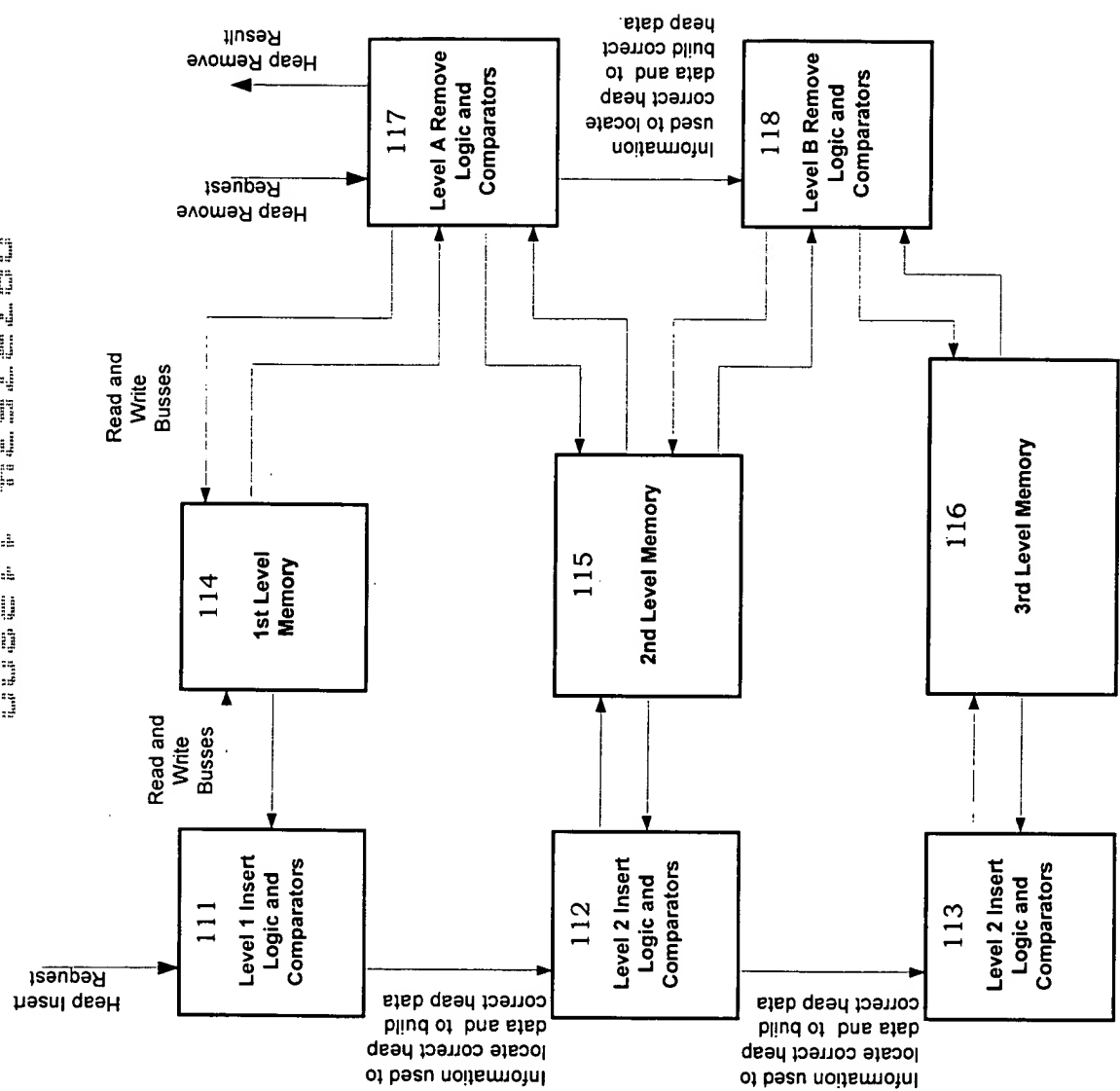


FIGURE 11

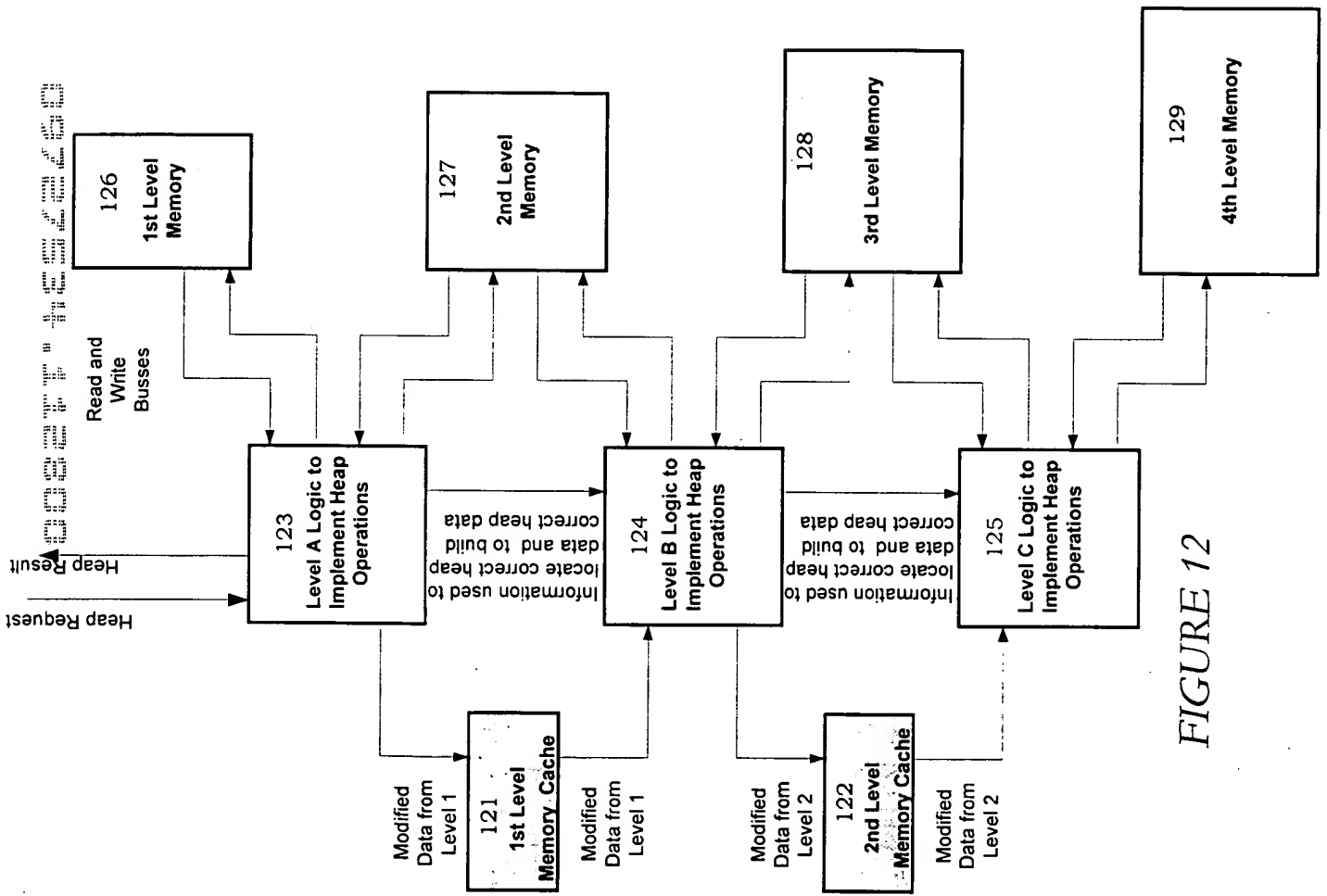


FIGURE 12

III. Request B checks the cache before it modifies the data that it read. If the cache indicates that B is operating on the same memory location in level 1 that request A just operated on, B uses the contents of the cache.

[illegible]

FIGURE 13